<P1-1>

Step1:

1) Search for “Choi” in the struct array and print it

Step2:

1) void print\_data\_in\_array(struct REGISTERATION\* data,int count)

Step3:

Store divided name in the array

Split the name by space

Search for “Choi” in the array

Compare the last name and print array

Print tag, date, fee-paid, name, age, organization, job

Step4:

void print\_data\_in\_array(struct REGISTERATION\* data,int count) {

int i,k;

for (i = 0; i < count; i++) {

/\*divide name\*/

if (strcmp(p\_name[1], "Choi") == 0 && p\_name[1]!=NULL) {

/\*print tag, date, fee-paid, name, age, organization, job\*/

}

}

}

<P1-2>

Step1:

1) Store the struct data in linked list

2) Search for “Choi” in the linked list and print it

Step2:

1) void make\_linked\_list(struct REGISTERATION\* reg, struct NODE\* head)

2) void print\_data\_in\_linked\_list(struct NODE\* head,int count)

Step3:

Store the data in linked list

Store divided name in the array

Split the name by space

Search for “Choi” in the linked list

Compare the last name and print the linked list

Print tag, date, fee-paid, name, age, organization, job

Step4:

void make\_linked\_list(struct REGISTERATION\* reg, struct NODE\* head) {

/\*set linked list\*/

if (p == NULL) {/\*first data insertion\*/

/\*Dynamic memory allocation\*/

/\*Copy the data and store in the linked list\*/

/\*connect to the next node\*/

}

while (i!=30) {

/\*data insertion\*/

/\*Dynamic memory allocation\*/

/\*Copy the data and store in the linked list\*/

/\*connect to the next node\*/

}

}

void print\_data\_in\_linked\_list(struct NODE\* head,int count) {

/\*set linked list\*/

for (i = 0; i < count; i++) {

/\*divide name\*/

if (strcmp(p\_name[1], "Choi") == 0 && p\_name[1]!=NULL) {

/\*print tag, date, fee-paid, name, age, organization, job\*/

}

/\*set node to next node\*/

}

}

<P8-1>

Step1:

1) Input recent data in the array

2) Create checksum with data

3) Write File with checksum

4) Read file with checksum and store in the array

5) Create another checksum with new data

6)compare two checksums

Step2:

1) void input\_recent\_data(char arr[][25],struct REGISTERATION\* re,int count)

2) void checksum(char s[][25],char\* check)

3) bool write\_file\_with\_checksum(const char\* fname,char data[][25],int count)

4) bool read\_file\_with\_checksum(const char\* fname, char data[][25], int\* num)

5) void checksum(char s[][25],char\* check)

6) void compare\_checksum(char ori[][25], char copy[][25])

Step3:

Store the recent data in the array

Bitwise operation

Create checksum with input data

Write file with checksum

Write tag, date, fee-paid, name, age, organization, job and add the checksum at the end of text

Read the file and store data in the array

Read file with checksum

Bitwise operation

Create another checksum with new input data

Compare two strings and print the result

Compare two checksums and confirm that two data are the same

Step4:

void input\_recent\_data(char arr[][25],struct REGISTERATION\* re,int count) {

for (int i = 10; i >=0; i--) {

/\*copy name to the array\*/

}

}

void checksum(char s[][25],char\* check) {

for (int k = 0; k < 10;k++) {

/\*make checksum for 10 values\*/

for (int k = 0; k < strlen(str); k++) {

/\*bitwise operation\*/

}

/\*convert integer to string\*/

/\*string concatenation\*/

}

/\*copy the checksum to the last of array\*/

}

bool write\_file\_with\_checksum(const char\* fname,char data[][25],int count) {

/\*open file\*/

if (pFile == NULL) {

/\*return false\*/

}

for (int i = 0; i < 11; i++) {

/\*write text file\*/

}

/\*close file and return true\*/

}

bool read\_file\_with\_checksum(const char\* fname, char data[][25], int\* num) {

/\*open file\*/

if (pFile == NULL) {

/\*return false\*/

}

while (fgets(str,30,pFile)) {

/\*copy data in the file to the array\*/

}

/\*close file and return true\*/

}

void checksum(char s[][25],char\* check) {

for (int k = 0; k < 10;k++) {

/\*make checksum for 10 values\*/

for (int k = 0; k < strlen(str); k++) {

/\*bitwise operation\*/

}

/\*convert integer to string\*/

/\*string concatenation\*/

}

/\*copy the checksum to the last of array\*/

}

void compare\_checksum(char ori[][25], char copy[][25]) {

/\*print two checksums\*/

if (strcmp(ori[10], copy[10]) == 0) {

/\*print two data’s checksum and two data are same\*/

}

else

/\*print two data’s checksum and two data are not same\*/

}